

Introduction to Neural Networks and their Applications in HEP

Marcin Wolter

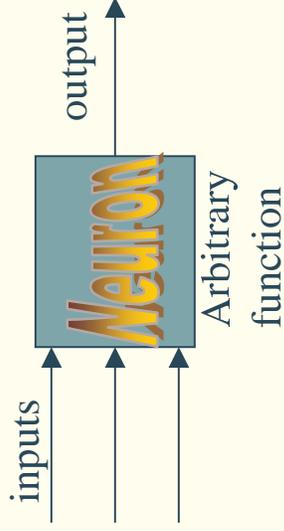
20 November 2000

Presentation is also on my WWW page: www.tufts.edu/~mwolte01



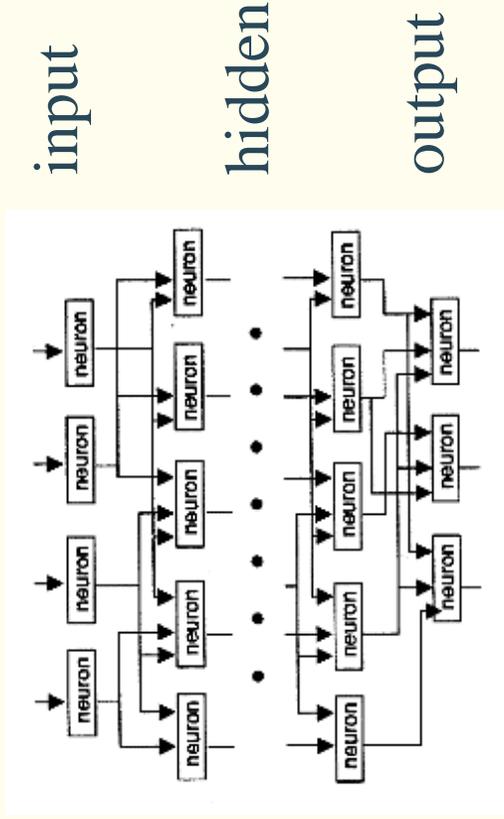
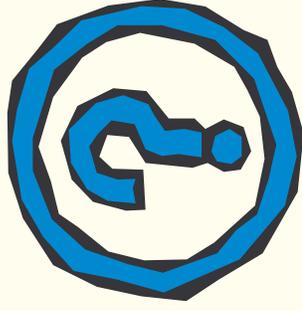
What is a Neural Network?

Simplified model of brain?



- Uses models of neurons
- Learns on examples
- Input noise resistant
- Resistant on cutting connections
- Parallel processing

Just an algorithm?

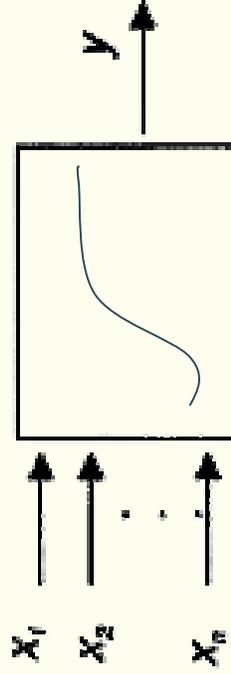


Where are NN used?



- # Expert systems (medicine, mechanics...).
- # Pattern recognition.
- # Predicting in meteorology, business (stock market).
- # In HEP:
 - data analysis (optimized selection)
 - triggering (hardware implementations-parallelism!!).

Neuron



Many inputs, one output

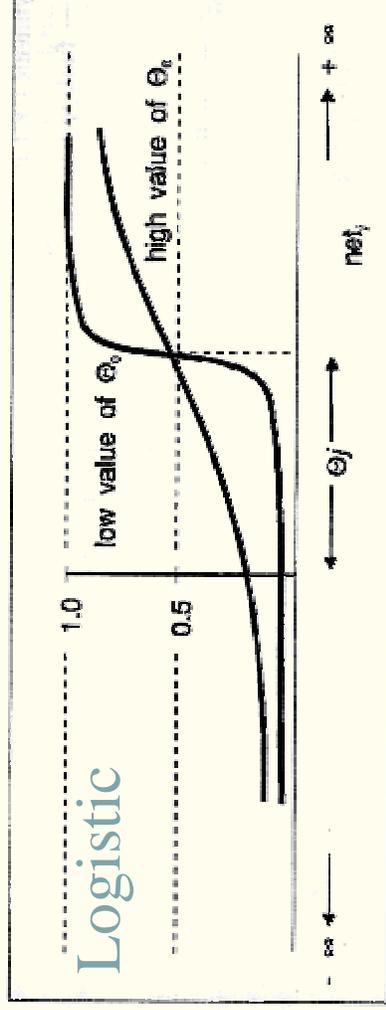
$y = f(w_1 \cdot x_1, \dots, w_n \cdot x_n)$

Where f is an arbitrary function: linear, step function or a very popular

logistic:

$$y = \frac{1}{1 + \exp(-\theta_0 \cdot \text{net}_j)}$$

$$\text{net}_j = \theta_j + \sum_{i=1}^n w_i \cdot x_i$$

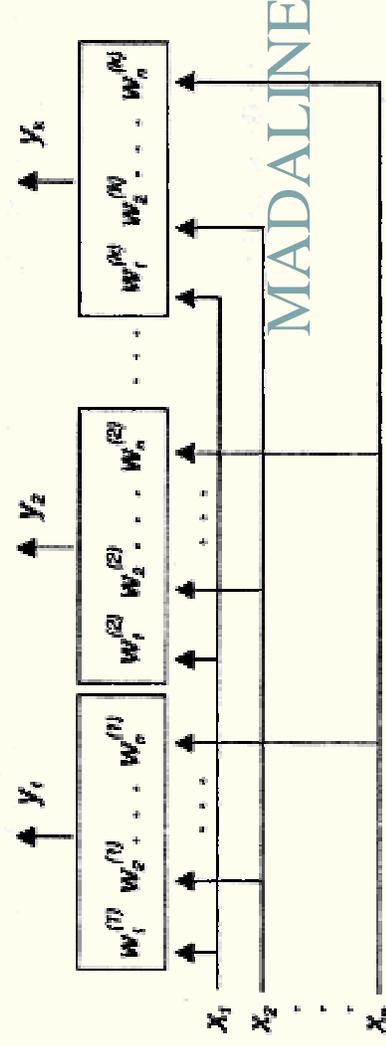
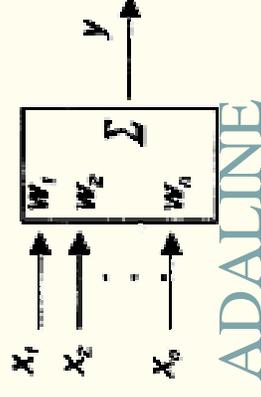


Linear networks

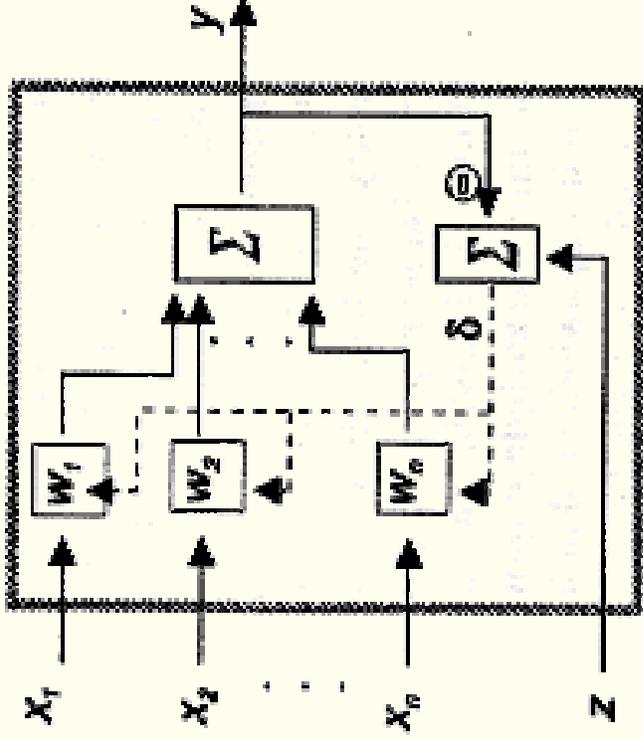
(Many) ADaptive Linear Network(s) –
(M)ADALINE

For every neuron: $y = \vec{W}^T \cdot \vec{X}$

For a neuron layer: $\vec{y} = W \cdot \vec{x}$



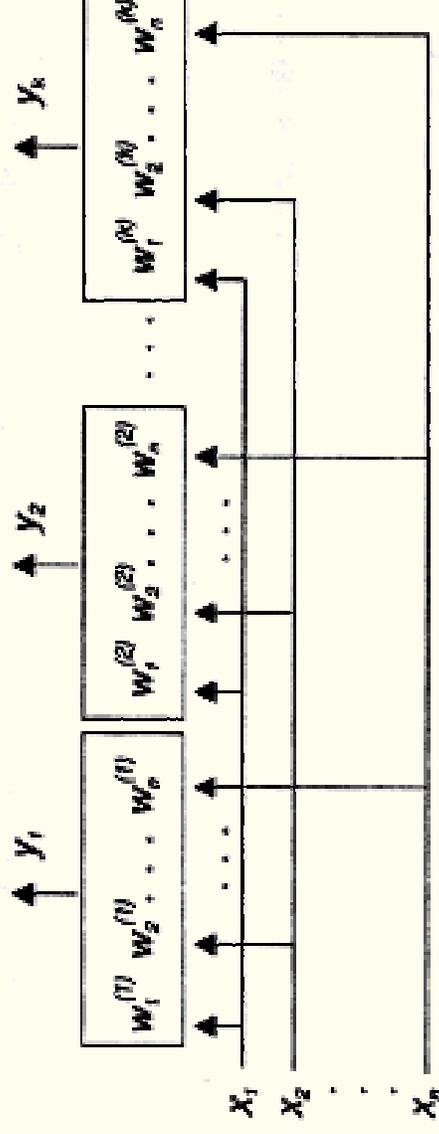
Single neuron learning



ADALINE

- X_i – inputs
- Y - output
- Z - proper output (learning!)
- TASK – minimize $\mathcal{K}^2 = \sum (z^{(j)} - y^{(j)})^2$
- Learning procedure:
- Calculate $\delta = z - y$
- And a new set of weights:
$$W' = W + \eta \cdot \delta \cdot X$$
- η - learning speed

Network learning



Same procedure, but now:

$$W^{(j+1)} = W^{(j)} + \eta(\vec{Z}^{(j)} - \vec{Y}^{(j)})(\vec{X}^{(j)})^T$$

where j denotes the learning step.

Unsupervised (Hebbian) learning

The network tries to classify objects basing on their similarity.

Method:
the given weight is increased proportionally to the product of the neuron input and the output.

$$w_i^{(m)(j+1)} = w_i^{(m)(j)} + \eta \cdot x_i^{(j)} y_i^{(j)}$$
$$y_i^{(j)} = \sum_k w_k^{(i)(j)} x_k^{(j)}$$

where m – neuron number, i – neuron input number.

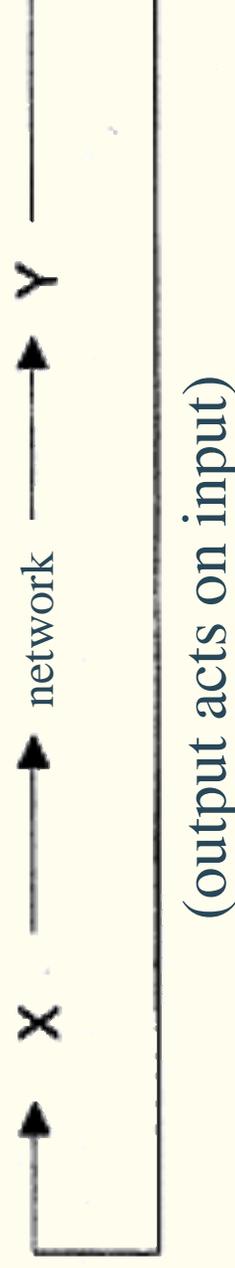
Increase the weights, where strong input gives a strong output signal (amplify the connections).

Hebbian learning

- # Input vectors are grouped into classes (but the “classes” are defined by a network and are not necessarily the classes we want).
- # Improved methods developed by Kohonen.
- # Unsupervised learning not frequently used in practical application.

Linear networks – a summary

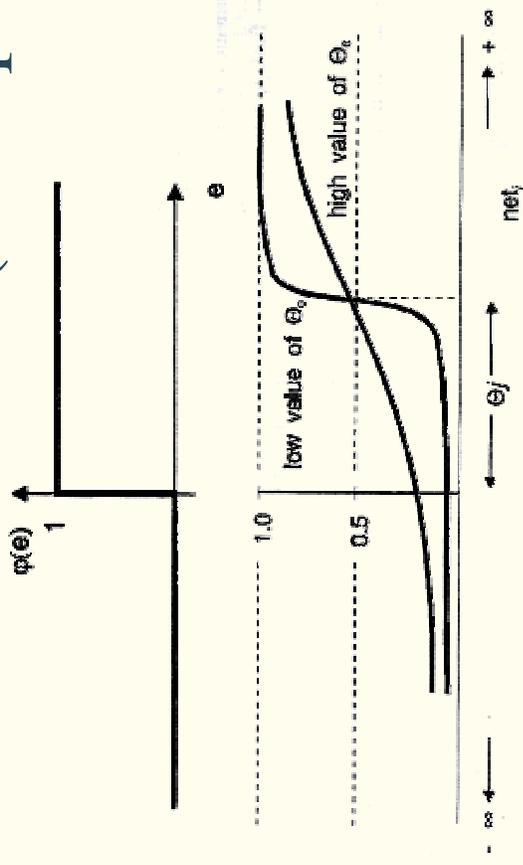
- # First widely used type of neural networks.
- # Limitation: only linear transformations (adding additional hidden layers does not improve the functionality).
- # Two types of networks (just a remark):
 - feedforward
 - autoassociative (Hopfield)



Non-linear networks

Non-linear activation functions (examples):

- step
- logistic

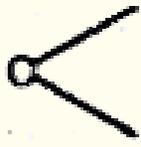
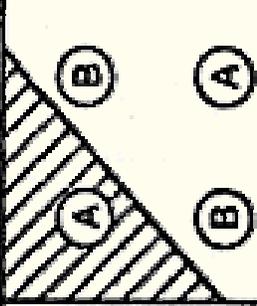
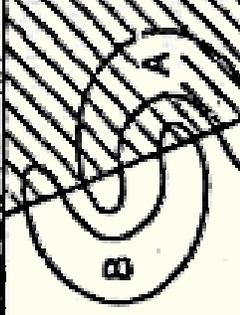
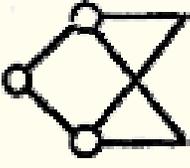
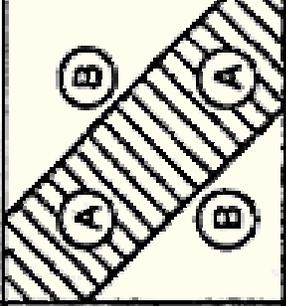
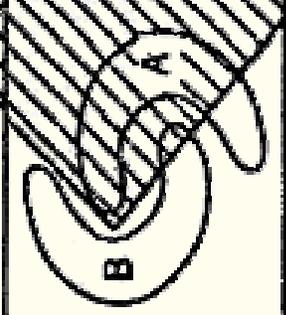
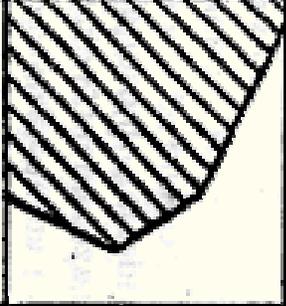
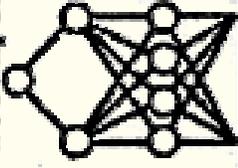
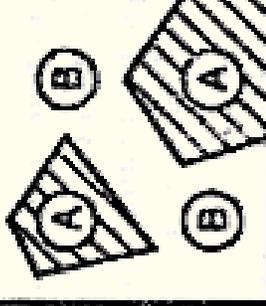
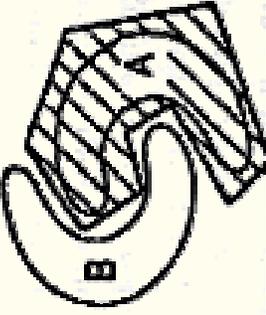
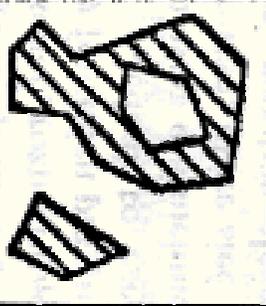


Network can perform any transformation.

A really powerful tool!!!

Performance of the network

(with a step activation function)

Structure	Type of Decision Regions	Exclusive-OR Problem	Classes with Mesned Regions	Most General Region Shapes
Single-layer 	Half plane bounded by hyperplane			
Two-layers 	Convex open or closed regions			
Three-layers 	Arbitrary (Complexity limited by number of nodes)			

one hidden layer

two hidden layers

Neural network performance

- # Single layer – obvious.
- # Two layers - 1st layer divides a “plane” by k straight lines, -2nd selects a region (simplex).
- # Three layers - 3rd layer joins simplexes building “accepted” or “rejected” regions (does not need to be one region).
- # Three layer network do “anything”, no need to add additional layers.
- # Complexity of the border between the regions depends on the number of nodes.
- # Logistic activation function smoothes the boundaries.

Learning - backward propagation

Single neuron – similar to linear neuron
(logistic function popular since easy to

differentiate): $y = \frac{1}{1 + \exp(-\beta e)}$

$$\frac{dy}{dx} = y(1 - y)$$

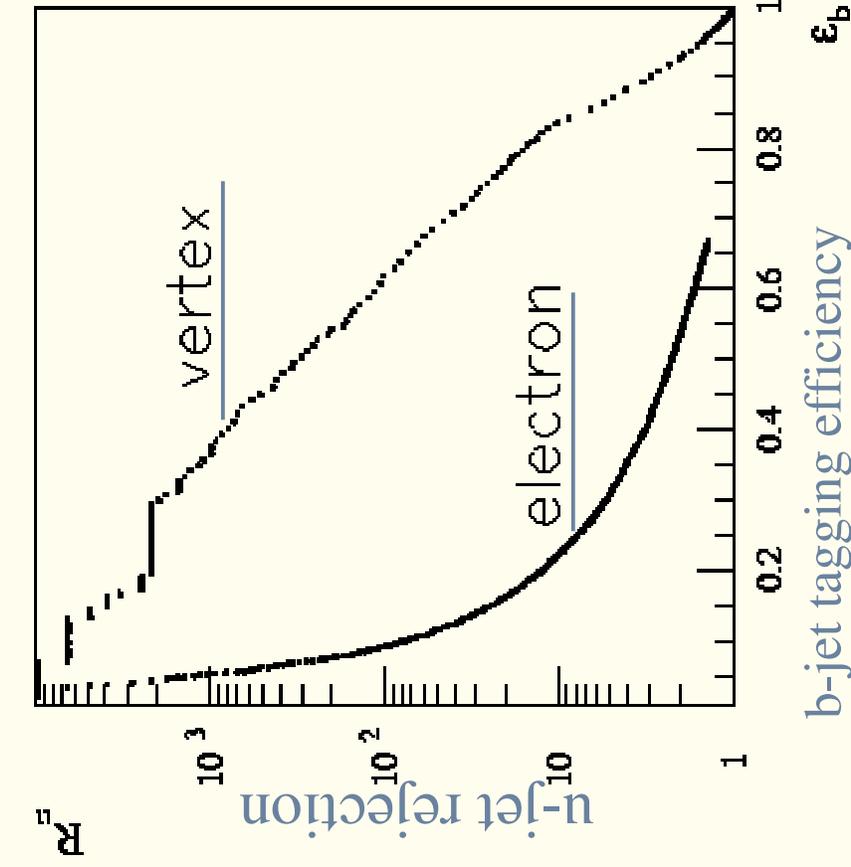
Many layers – a **backpropagation** method.
An error δ is propagated back through the
net using the present weights (“revolution”
in neural networks in middle 80’).

Neural networks in HEP

- # **Trigger** – hardware implementations (fast, parallel processing) – CPLEAR, H1, DIRAC.
- # **Offline** – data selection (NN simulators), ability to make efficient selection in multidimensional space.

Data selection problem – an example from ATLAS

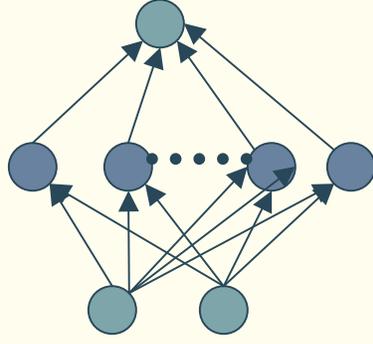
A. Kaczmarska, M.W. ATLAS-INDET-2000-023



- # Combining two b -jet tagging methods based on soft electron and long decay path of b mesons.
- # $\text{BR}(b \rightarrow e) \approx 17\%$ only!
- # Combination of “good” and “poor” methods.
- # Background: μ -, c - and g -jets.

Designing a neural network

Two inputs, one output.

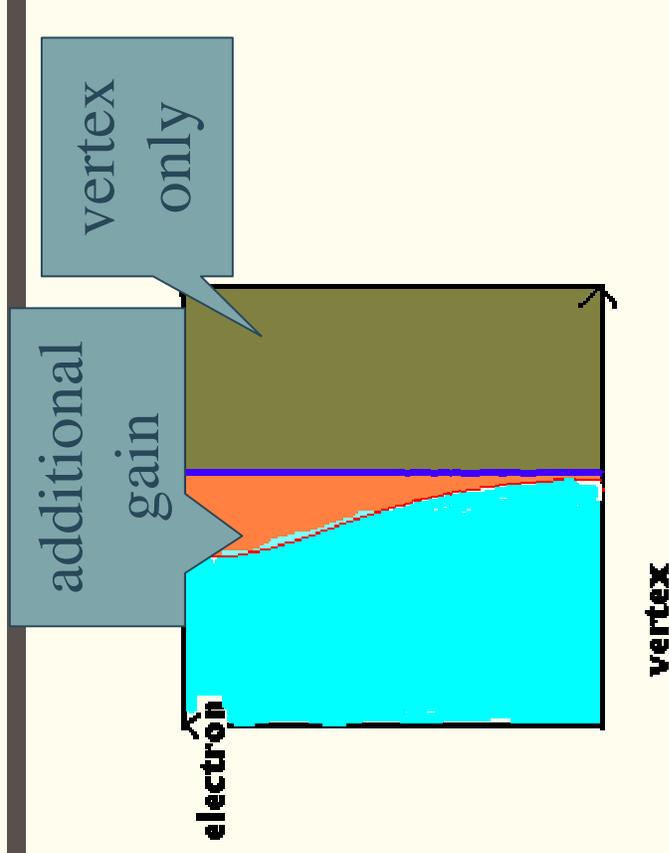


One hidden layer.

Smooth output function – activation logistic.

Simple feedforward network

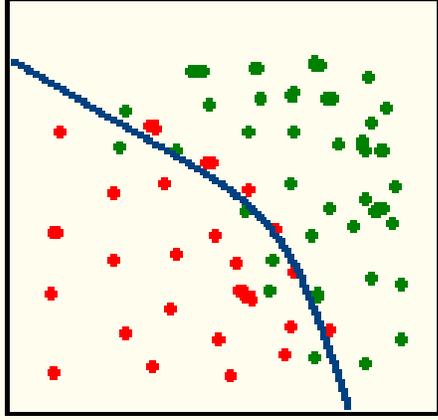
Stuttgart Neural Network Simulator used.



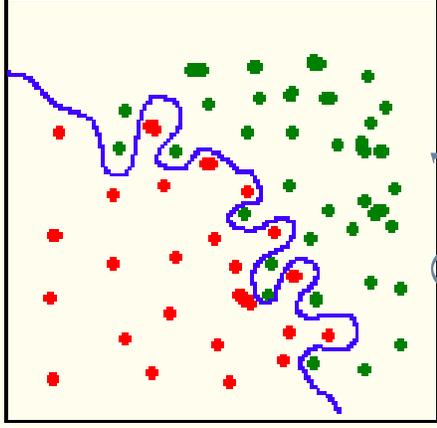
Learning procedure

- # Training sample: b -jets (signal), u -, c -, g -jets (background).
- # Verification sample: same proportions of signal and background, but independent data sets.
- # χ^2 for both samples verified during learning process to avoid *overlearning*.

Overlearning

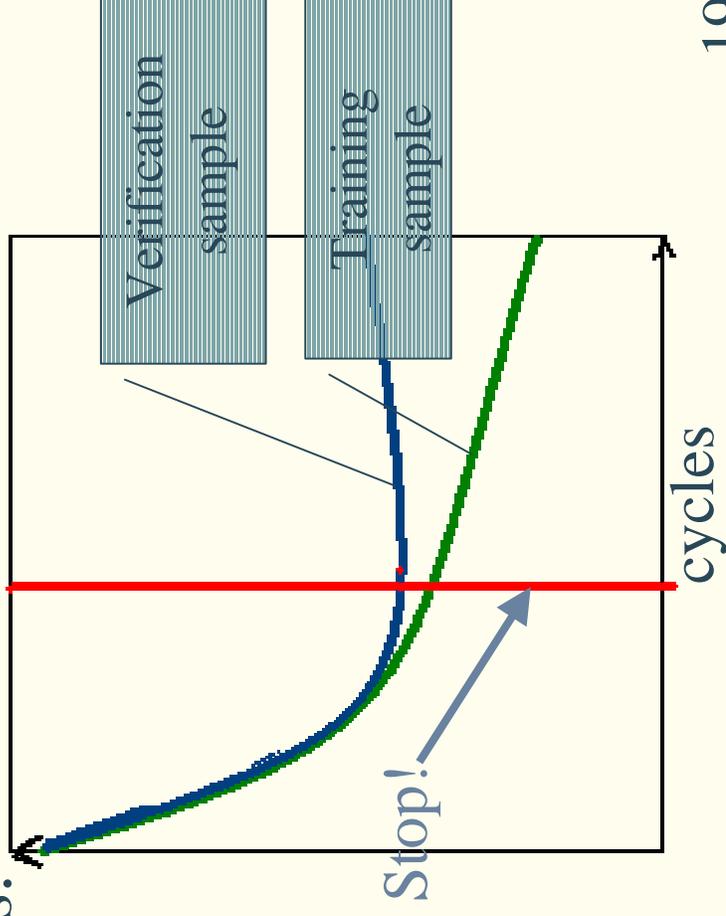


Proper

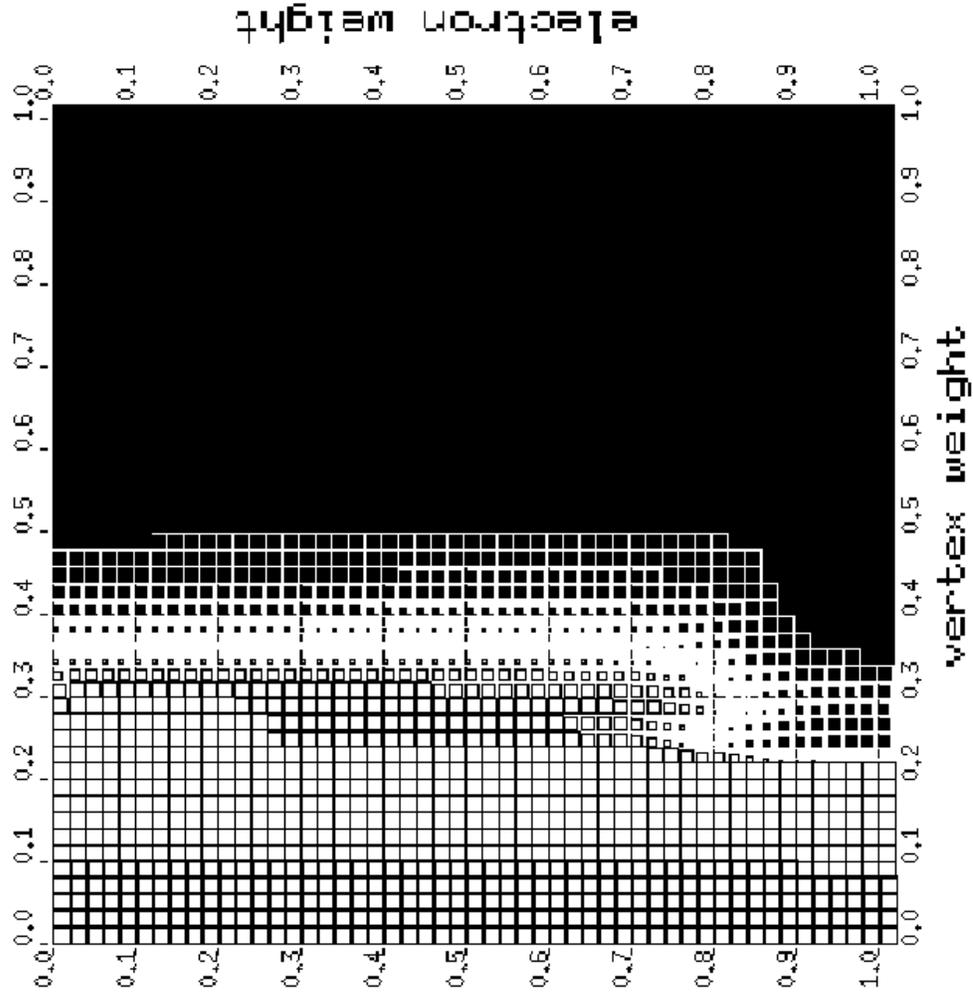


Overlearned

Overlearning - network learns single events instead of general rules.

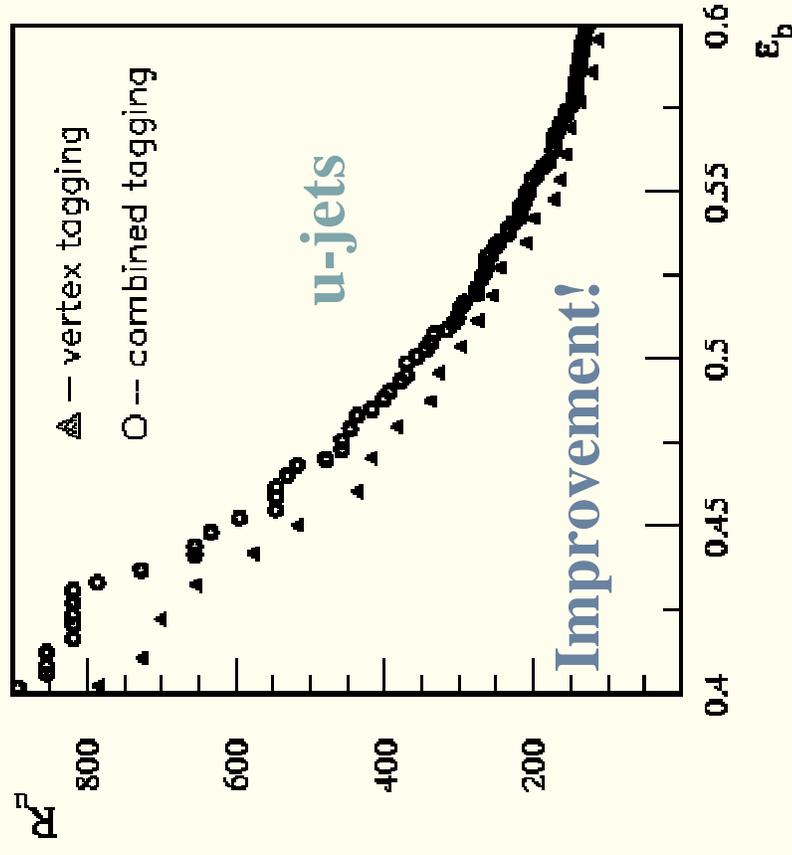
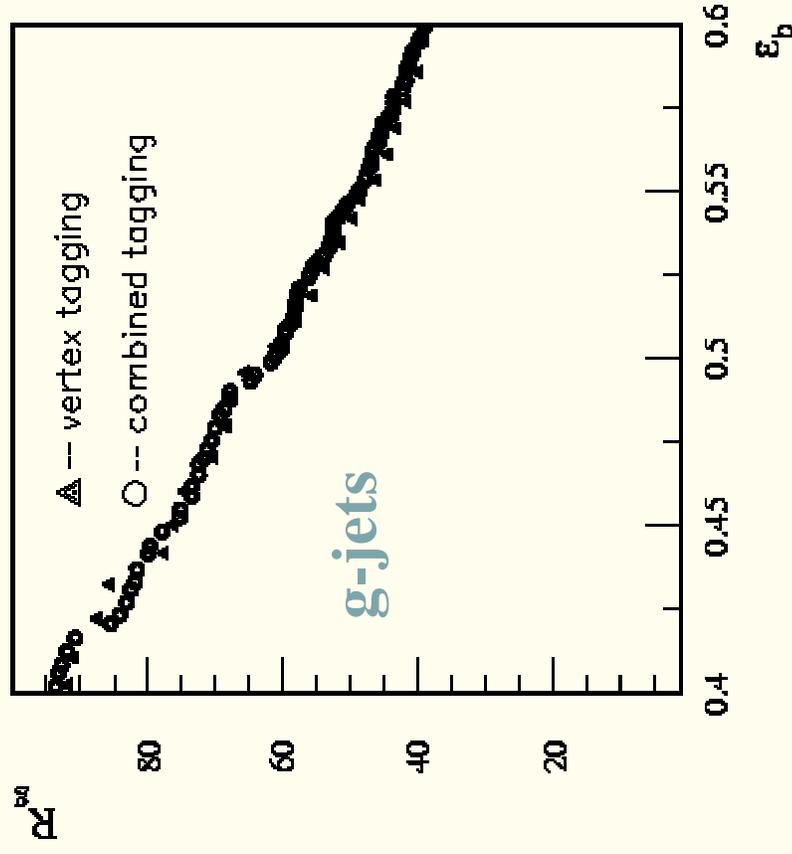


Weight function

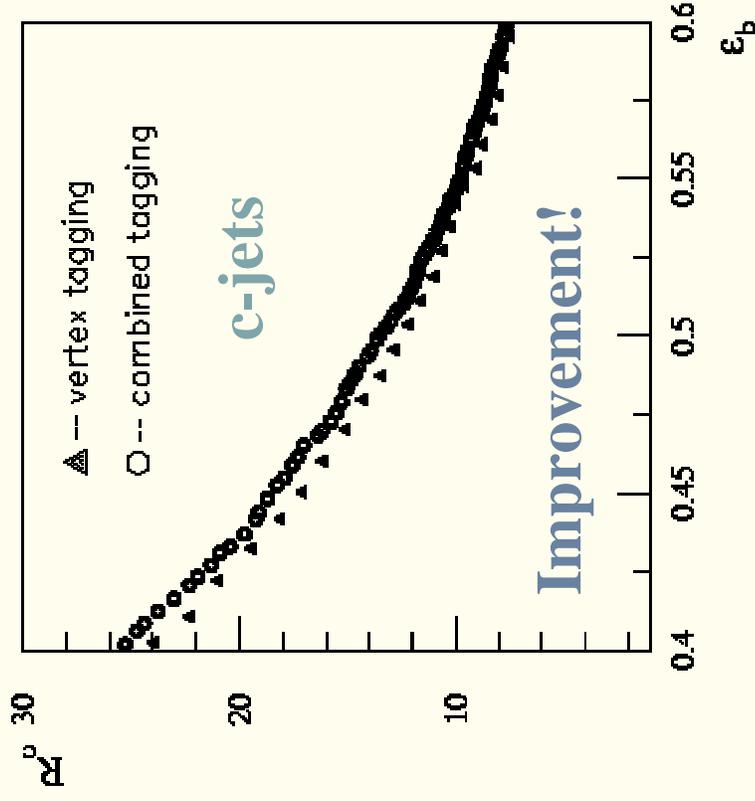


Function shape agrees with out intuition.

Improvement of background rejection



Improvement of background rejection



- # Improved rejection against u - and c -jets.
- # Network can be specially trained against g-jets (but worse rejection against another background sources).
- # NN allows to combine two tagging methods.

Conclusions

Neural networks:

- # learn on examples,
- # robust (input noise, “cutting” connections),
- # naturally parallel (fast when hardware implemented),
- # allow to find optimal selection criteria (using examples).

A useful tool for event selection (offline and online).